Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 190 (2021) 690–694

Procedia Computer Science

www.elsevier.com/locate/procedia

2020 Annual International Conference on Brain-Inspired Cognitive Architectures for Artificial Intelligence: Eleventh Annual Meeting of the BICA Society

Lightweight Surround View Algorithm for Embedded TDA3xx Platform

Andrejs Rudzitis[a,*], Margarita A. Zaeva[a]

[a] *National Research Nuclear University MEPhI (Moscow Engineering Physics Institute), Kashirskoe shosse, 31, Moscow 115409, Russian Federation*

Abstract

Currently, the direction of computer technology in the automotive industry is developing strongly. One of the newest areas is the driver assistance system. One of its main functions is the all-round visibility. It allows you to create a view of the car from above, and it is possible to display from different viewing angles using images from four cameras placed on all sides of the car. There are a large number of solutions in this area, but they are usually specific to different types of equipment. In this paper we will consider the Circular view algorithm on the embedded tda3xx Board.

Keywords: surround view; ADAS; lens distortion correction; birds-eye view; calibrations

1. Introduction

There are multiple solutions for birds eye view and lens distortion correction implementation ([1] and [2]). Main goal in all of them is to create a lookup table for each pixel in an image mapped to original image pixel. Mentioned studies use mainly OpenCV (see [3] and [4]) or similar algorithms for camera calibration. In case of image stitching Alpha Blending [5] or specialized blending formulas (see [6]), or invariant features (in [7]) are used. There is a direct project from Texas Instruments [8], that utilizes specialized TDA3 resources but requires a lot of technical data about cameras and lenses, which is usually not provided by hardware manufacturers. External calibration requires additional software and specially trained staff. These features must be taken into account when building Advanced Driver Assistance Systems (ADAS) ([9], [10]).

---

* Corresponding author. Tel.: +7-495-788-5699.
E-mail address: andreyrud@inbox.lv

10.1016/j.procs.2021.06.080

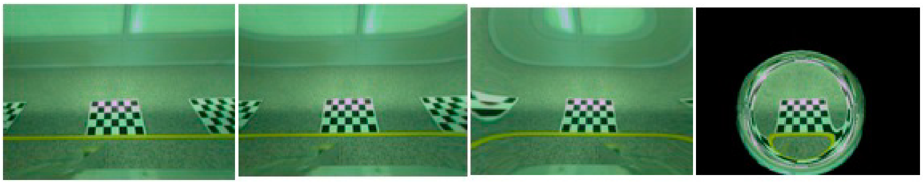

Fig. 1. Lens distortion correction examples using different coefficient calculation methods.

1.1. TDA3xx features and resources

TDA3xx platform is specialized to work with video data on vehicles. Cortex M4 is used to control resources, usually algorithms are not implemented on it because of it low performance. All specialized resources are configured using this central processing unit (CPU). One of the main TDA3xx advantages its Imaging Sub System (ISS). It allows performing basic image operations bypassing any calculations on other processors. It is a key component for our algorithm development. ISS consists of three modules. Camera Abstraction Layer (CAL), this module is responsible for data transfer from sensor to board. Image Signal Processor (ISP), allows basic transformations such as white balance, auto exposure and de-bayering. Still Image Coprocessor (SIMCOP) is responsible for Lens Distortion Correction (LDC) and for Inverse Perspective Mapping (IPM). First function is implemented using lookup tables (LUT) and second using homography matrices. Simultaneous usage of these functions is not advised because it leads to artifacts on image. Algorithms calculation on TDA3xx are usually performed on digital signal processor (DSP) c66x or EVE vector coprocessor. In our task only single DSP was used which may lead to price decrease due to exclusion of vector coprocessors. DSP c66x processors, though not very powerful (600 MHz frequency and poorly perform on floating point numbers have several features such as ability to perform eight multiplication in single operation and load balancing between two arithmetic blocks). For successful parallelization, several conditions must be met. These restrictions were heavily considered during development of our algorithm.

2. Proposed solution

Surround view algorithms consists of three parts:

- External and internal parameter acquisition or calibration.
- Transformation construction for birds eye view and wide-angle lens barrel effect correction.
- Stitching four transformed images into one.

2.1. Calibration

To implement correct birds eye view image it is necessary to determine camera intrinsic parameters, such as focal length, resolution and distortion coefficients. These parameters are mainly needed to correct wide-angle lens barrel effect that are used for field of view intersection with neighboring cameras. These parameters cant be easily acquired externally, and several experiments must be made, to provide sufficient data for OpenCV calibration library [11]. This solution has several disadvantages that do not allow using it in already established car manufacturing process. This calibration assumes creation of multiple chessboard snapshots under different points of view and requires some knowledge in optics. This method is very difficult to automate and shorten. We found that solution that is more acceptable would be calibrating single lens and further usage of these parameters in all future installments. Obvious advantage would be lack of intrinsic calibration and therefore no need of addition software and experts, but in this case, image quality would we significantly less (Fig. 1).

Where first image on the left is the result of multiple image calibration and second calibration with one image, but with predefined distortion coefficients (second on the left) are practically identical, on the other hand single image calibration produces significant errors on image edges. They are a vital area where all images are connected with each other.

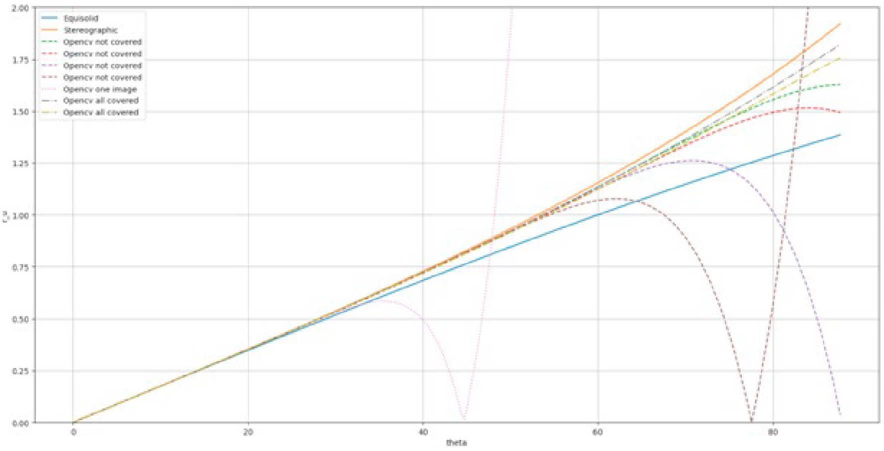

Fig. 2. Lens distortion correction examples using different coefficient calculation methods.

We determined that some parameters, such as distortion coefficients, have insignificant effect on a resulting image. On the other hand, parameters like optical centers can change drastically (Fig. 2). To calculate optical center parameter one image of chessboard from each camera is sufficient. This approach can be easily automatized.

Extrinsic camera parameters can be obtained using the same single shot used for intrinsic calibration. Thus, all necessary parameters are calibrated using the same technological process.

2.2. Transformation or lookup table acquisition

Inverse perspective mapping calculation and barrel distortion correction are highly demanding operations, but TDA3 has a special hardaware solution SIMCOP that is specially designed to perform this kind of tasks. First task is implemented with help of extrinsic camera parameters and homography matrix. Lookup table that contains disposition of every pixel on an image corrects lens distortion that causes image to have straight lines where they are supposed to be straight.

Using both function of four cameras causes an efficiency drop and produces earlier mentioned artifacts. This problem led to creation of a special lookup table that both performed inverse perspective mapping and lens distortion correction. For this operation, it is required to convert the homography matrix into lookup table and then combine them into a single table. Let there be a lookup table with dispositions (dx_i, dy_i) for each image pixel to correct distortion. For inverse perspective mapping there is matrix H.

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix} \tag{1}$$

Where new coordinates $(\overline{x_i}, \overline{y_i})$ for each pixel are calculated using formula

$$\begin{pmatrix} x_i' \\ y_i' \\ z_i' \end{pmatrix} = H * \begin{pmatrix} x_i + dx_i \\ y_i + dy_i \\ 1 \end{pmatrix}, \overline{x_i} = \frac{x_i'}{z_i'} \overline{y_i} = \frac{y_i'}{z_i'} \tag{2}$$

To convert result back into a lookup table we perform simple subtraction:

$$(\overline{dx_i}, \overline{dy_i}) = (\overline{x_i} - x_i, \overline{y_i} - y_i) \tag{3}$$

2.3. Image stitching

We decided to use Alpha Blending algorithm that allows to combine images with gradual transition. We decided to implement this task on a DSP C66x core but due to its poor performance with floating point numbers. For example, floating-point number addition and multiplication can take from two to six CPU cycles in the meantime similar integer operations can be performed up to 8 times in one cycle. This lead to modification in original algorithm. All coefficients were converted to fixed-point format Q8. After that, we constructed a mask that contained all coefficients for every pixel of composite image. For optimal performance on DSP it is necessary to exclude all branching paths. This led to storage of all image pixels in mask and unconditioned traverse over all image. All these optimizations resulted in performance increase from 1 fps to 25 fps.

Conclusions

Developed calibration algorithm can be automated without presence of any human. Resulting bird's eye view image construction was fully transferred to hardware solutions, which lead to decrease of load on other cores. Image stitching algorithm was optimized for DSP cores and increased performance up to 25 times.

Current goals are to increase image resolution by optimizing work of SIMCOP. It is planned to achieve it by combining images before passing them through hardware. In addition, it is planned the extending sensor count up to eight for city articulated buses. Main issues are increased load due to camera count and bus deformation on turns.

Acknowledgements

This work was supported by Competitiveness Growth Program of the Federal Autonomous Educational Institution of Higher Professional Education National Research Nuclear University MEPhI (Moscow Engineering Physics Institute).

References

[1] Okutsu Ryota, Terabayashi Kenji, Aragaki Yohei, Shimomura Noriko and Umeda Kazunori. (2009) "Generation of overhead view images by estimating intrinsic and extrinsic camera parameters of multiple fish-eye cameras." *Proc. 11th IAPR Conf. Mach. Vis. Appl. MVA* **5** (5) : 447-450.

[2] Devernay Frederic and Faugeras Olivier. (2001) "Straight Lines Have to Be Straight Automatic Calibration and Removal of Distortion from Scenes of Structured Environments." *Mach Vis Appl.***13** (1): 14-24.

[3] Shao Xuan, Liu Xiao, Zhang Lin, Zhao Shengjie, Shen Ying and Yang Yukai. (2019) "Revisit Surround-view Camera System Calibration." *IEEE Int. Conf. Multimed. Expo* **2019-July**: 1486–1491, `doi:10.1109/ICME.2019.00257`.

[4] Li Bo, Heng Lionel, Koser Kevin and Pollefeys Marc. (2013) "A multiple-camera system calibration toolbox using a feature descriptor-based calibration pattern." *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, Japan, 1301–1307, `doi:10.1109/IROS.2013.6696517`.

[5] Thomas Porter and Tom Duff. (1984) "Compositing digital images." *SIGGRAPH Comput. Graph* **18** (3): 253-259`doi:10.1145/964965.808606`.

[6] Liu Yu-Chih, Lin Kai-Ying and Chen Yong-Sheng. (2008) "Birds-Eye View Vision System for Vehicle Surrounding Monitoring." **4931** 207–218, `doi:10.1007/978-3-540-78157-8_16`.

[7] Brown Matthew and Lowe David. (2007) "Automatic Panoramic Image Stitching using Invariant Features." *International Journal of Computer Vision* **74**: 59–73, `doi:10.1007/s11263-006-0002-3`.

[8] Nagpal Raghav, Krishna Chaitanya, Jayababu Vijaya, Ragavan R. Navinprashath, Bhat Radhesh and Ghosh Dipanjan. (2019) "Real-time traffic sign recognition using deep network for embedded platforms." *Electronic Imaging* **33** (1), `doi:10.2352/ISSN.2470-1173.2019.15.AVM-033`.

[9] Rybina Galina and Fontalina Elena. (2019) “Automated Acquisition, Representation and Processing of Temporal Knowledge in Dynamic Intelligent Systems.” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*, `doi:10.1007/978-3-030-33274-7_9`.

[10] Sergey Yu. Misyurin, German V. Kreinin, Natalia Yu. Nosova and Andrey P. Nelyubin. (2021) “Kinematics and Dynamics of the Spider-Robot Mechanism, Motion Optimization. *Brain-Inspired Cognitive Architectures for Artificial Intelligence: BICA*AI 2020. BICA 2020. Advances in Intelligent Systems and Computing* **1310**, Springer, Cham, `doi:10.1007/978-3-030-65596-9_38`

[11] “OpenCV: Camera Calibration. [Online]. `https://docs.opencv.org/master/dc/dbb/tutorial_py_calibration.html`Accessed: 06-Apr-2020.